1995

An Evaluation of Transmitting Compressed Images in a Wide Area Network

Melliyal Annamalai

Bharat Bhargava
Purdue University, bb@cs.purdue.edu

Report Number:
95-064

AN EVALUATION OF TRANSMITTING COMPRESSED IMAGES IN A WIDE AREA NETWORK

Melliyal Annamali
Bharat Bhargava

Department of Computer Science
Purdue University
West Lafayette, IN 47929

CSD TR-95-064
October 1995

An Evaluation of Transmitting Compressed Images in a Wide Area Network

Melliyal Annamalai and Bharat Bhargava

Abstract

Retrieving large data objects (of the order of hundreds of thousands of kilobytes) in a wide area network such as the Internet takes several seconds and occasionally minutes. As a step towards reducing response time we have studied the possibility of reducing the size of the data object by "losing" some of the data and then retrieving it. We have focussed on transmission of images across a wide area network. Images are amenable to losing data without losing the semantics of the image. Lossy compression techniques result in a smaller size, lower quality image. We have studied how the quality of the image can be sacrificed to achieve higher speed in retrieval over a wide area network. We have identified parameters involved in the retrieval process and conducted experiments to see how they can be varied to speed up retrieval. We have developed a scheme to reduce response time at the expense of the quality of the object depending on the current values of the parameters.

1 Introduction

Digital libraries provide online access to a vast number of distributed text and multimedia information sources in an integrated manner. Digital libraries encompass the technology of storing and accessing data, processing, retrieval, compilation and display of data, data mining of large information repositories such as video, audio libraries, management and effective use of multimedia databases, intelligent retrieval, user interfaces and networking. Digital library data include texts, figures, photographs, sound, video, films, slides, etc. The sizes of the data and information repositories available are enormous. NASA collects terabytes of earth and space science data every day. DOD has a repository of all the data on all events of the Gulf War. Almost every organization has repositories of old versions of software and business related data. The Global Information Infrastructure will bring about the integration, management, and communication of these gigabytes of data in a parallel and distributed environment over national and international networks. Such an infrastructure will have a major impact on people in all walks of life – students, businessmen, researchers, engineers, government employees, and scientists. Advances in databases and networking are required to succeed in the technical program of building an environment where everyone in the world will have access to distributed stores of information at a reasonable cost.

The goal of this work is to present a method to reduce the response time. Many researchers have worked on the problem of reducing response time when images are retrieved in a wide area network. Thumbnail images for browsing is one solution which is provided by many WWW clients. "Losing" some of the details of the image to reduce the size of the image file is another solution. The reduced file size leads to a decrease in the response time. We have studied the factors involved in reducing the quality of the image and reducing response time.

We studied large data objects containing images in the context of a multi-resolution data model as the one proposed by [7]. Images can be stored, manipulated and viewed in multiple resolutions. Images are rich in semantic content and careful manipulation of the image will result in an image so that no visible information lost. Lossy compression techniques such as JPEG exploit this fact and reduce the number of bits used for chrominance and retain the number of bits used for luminance for example, since human eyes are more sensitive to luminance than chrominance. "Losing" some of the chrominance information does not result in any visible semantic loss in the image but will result in a smaller size file which will be easier to manipulate and transmit. Even a 10% saving in file size will result in the saving of a couple of seconds during transmission. While retrieving a large number of images in succession, these seconds add up to a significant saving. The trade-off between quality and time should be decided by taking into account various parameters. The parameters we have identified are user level of interest (quality level), conduciveness of the image to compression without losing semantic content, distance over which the image is retrieved, time of day of transmission, computation power both at the sender and receiver sites, and the physical media over which the image will be retrieved. Some of these parameters can be traded off to achieve a lower response time. Quality of the image can be reduced to reduce the size of the image which will in turn lead to the reduction of the transmission time. But the lossy compression techniques used to lose information in an image will increase the access component of the response time. The time taken to compress the image depends on several factors like the compression technique used and the attributes of the image.

2 Related Work on Reducing Image Transmission Time

Several researchers have worked on the problem of reducing the response time for retrieving images over a network. The methods range from reducing the image size to progressive transmission.

2.1 Thumbnail Images

Very often when users are browsing images in a distributed database system, they look at many images before they pick one that they want to view in detail or more closely. Retrieving the images the user browses through is very time consuming. Popular WWW browsers like Mosaic and Netscape offer the option of setting up an index page of thumbnail images. A thumbnail image is a small (with typically 80x80 resolution) version of the original image useful in helping the user decide whether to retrieve the full image. The filesize of the thumbnail image is typically about 2% that of the original.

2.2 Progressive Transmission of Images

Knowlton proposed a scheme for the lossless progressive transmission of images over low bandwidth lines such as telephone lines [6]. This scheme was later extended by Hill et al for rich imagery found in remote sensing applications [4]. The approach is based on sending coarse information first, to inform the user early in the reception phase about the general nature of the image. Images are encoded so that during transmission the entire display shows a rough

version of the image in 'fat pixels' [4]. If the user wishes to see more detail, additional data is sent and used to refine these pixels, until the exact original image is seen. At any time during this transmission, the user can abort the session. Examples show that the user may be able to make that decision after only 2% of the image has been received.

2.3 Image Rendering by Adaptive Refinement

The objective of the work by Bergman et al [1] is also to convey as much information to the user as soon as possible. But the method is different and focusses on the use rendering procedure. The renderer developed as part of this work constantly refines the image to improve it and is also adaptive meaning that the improvement adapts to the particular nature of the image - the polygons and pixel locations whose further processing is likely to make the greatest improvement in the picture quality. From the user's perspective, the quality of the image from a standard renderer improves with time. The adaptive rendering system proceeds as follows: vertex display, edge display, flat shading, shadow display, gouraud shading, phong shading, and anti-aliasing.

2.4 The Wisconsin Project

NASA has awarded a $500,000 grant to the University of Wisconsin, Madison, and the Space Telescope Science Institute, Baltimore. The team is working on improving the rate at which large digital images can be transferred across the network.

3 Image Formats

There are a wide variety of image file formats in use now for image processing application [5]. Choosing a particular format for a given application generally involves several interdependent considerations: quality, flexibility, computation, storage, transmission efficiency, and support by existing programs. Some formats like vector-based ones are efficient with respect to storage but require additional computing power and software development to encode and decode [5]. Similarly, image compression techniques reduce storage and transmission requirements at the expense of greater computing time. Our work uses the JPEG compression format as a case study. We also use GIF a format to compare JPEG with respect to encoding and decoding time. The following subsections contain a brief description of GIF and JPEG formats.

3.1 Graphics Interchange Format (GIF)

Graphics Interchange Format is a protocol for communicating raster graphics data [5, 8]. GIF is a simple format designed primarily as a transmission format for a data stream than as a storage format for an image file. This is the reason behind the sequential organization of the format. A GIF file consists of the header block, the logical screen descriptor block, an optional global color-table block, blocks of image data or special-purpose blocks, and the trailer block which is a terminating code.

A GIF encoder creates the header block for the input files and appends the input files to the header block. The header block contains information about the color table, dimension and resolution of the image, and initialization values for hardware parameters needed to render the

graphics image. The decoder initializes device setup parameters, loads the color table, reads in data files, and displays the image.

3.2 JPEG Image Compression Technique

JPEG is a standardized image compression mechanism [3]. JPEG stands for Joint Photographic Experts Group, the original name of the committee that wrote the standard. JPEG is designed for compressing either full-color or gray-scale images of natural, real-world scenes. It works well on photographs, naturalistic artwork and similar material but not so well on lettering, simple cartoons, or line drawings. GIF (Graphics Interchange Format) performs better for these images.

JPEG is a lossy compression technique [5, 8]. It does not reconstruct the original image bit-for-bit but reconstructs an image which looks very similar to the original to the human eye. JPEG primarily stores information on color changes, particularly variations in brightness, because the eye is very sensitive to them [5]. One can choose the extent of compression while choosing JPEG. The extent of compression decides how much data is lost. We use percentages to denote the amount of data lost. A 10% JPEG file is one which has retained 10% of the original file. This does not necessarily mean that the file size is 10% the original size. JPEG compresses some files better than others (better ratio with respect to size) depending on the content of the image. For example an image file with predominantly one color with many shades compressed (at 10% quality) to a file with size 48.15% of the original. Another image file with many colors compressed to a file with size 14.59% of the original. 75% quality is perfect for human eyes in most cases. 10%, 30%, 50%, 75% etc. are referred to as levels of quality in this paper.

The JPEG compression and decompression procedures are more complex than the GIF encode and decode procedures. The JPEG compression steps are [5, 8]:

- *Subsampling:* Adjacent pixels are combined into a single value. In a color image, JPEG compresses each color component separately. It applies compression to color data expressed as luminance (brightness) and chrominance. Since the human eye is less sensitive to color changes than to brightness changes, chrominance channels can be coded with more loss than the luminance channel.

- *DCT Coding:* Discrete Cosine Transform is applied to convert raster data into rate-of-change information. DCT turns an array of intensity data into an array of frequency data that tell how fast the intensities vary. JPEG applies DCT to 8x8 rectangles of pixel data. The result of the DCT is a set of spatial frequencies, $F(u, v)$, which indicate to what degree the values change at each of a set of rates. The point of changing from intensities to frequencies is that slow changes are much more noticeable than fast ones, so the low-frequency data is more important than high-frequency data for reconstructing the image. DCT is reversible using reverse DCT.

- *Quantization:* This step truncates the results of DCT coding to a smaller range of values. This is the lossy step and the quantization coefficients decide how much data is lost and hence the extent of compression. JPEG uses linear quantization, that is, each of the DCT values is divided by a quantization factor and rounded to an integer to get the value that

will be stored. An 8x8 table of quantization factors is used, one for each output term. While decoding, the original values are approximately recovered by multiplying by the quantization factor.

- *Huffman or Arithmetic Coding:* The results of quantization are compressed using either one of the coding schemes. Most current implementations use Huffman, since that is public domain.

3.3 Suitability of JPEG for Multi-resolution Data Type

A lower resolution data object will consume less time during data transmission because the size is almost always smaller than the higher resolution data object. If a lower resolution data object is used, then quality is being traded off for response time. Several researchers have come up with progressive techniques where the image is transmitted step by step [4]. The data sent in each step enhances the data sent in the previous step. The response time for the user is low since the first low resolution image is small in size and also the user can start viewing something while the rest of the data is received instead of waiting for the whole data object to be received. The quality levels of JPEG, as defined by us are not conducive to progressive transmission since one quality level cannot be used to enhance a lower quality level. It is a new, complete image in itself and can replace the old lower quality display but cannot add to it. Progressive JPEG techniques are expected to make an appearance very soon [3].

3.4 Image Compression Techniques

Image compression can be defined as a method which maps an original image raster into a bit stream suitable for communication over or storage in a digital medium so that the number of bits required to represent the coded image is smaller than that required for the original image [2]. Image compression techniques aim at minimizing storage space and/or communication time. We are concerned with how communication time can be minimized using image compression. Compressed images can be squeezed into smaller bandwidth and one can have more video channels on fiber networks [2]. Some compression techniques allow progressive transmission, providing increasing good images starting from a coarse image and improving it as new bits arrive. This is very useful for browsing. Bandwidth is becoming faster and cheaper due to current advances in network technology, but there will always be users with low capacity links such as from hand-held communication devices and low cost modems.

There are two basic types of compression: *lossless* and *lossy* compression. In lossless compression, the original image can be perfectly recovered from the compressed representation [2]. Compression ratios rarely cross 4:1 [2]. In lossy compression, the original pixel intensities cannot be perfectly recovered. Lossy compression techniques attempt to remove redundant information. Typical compression ratios range from 4:1 to 32:1 [2]. Higher compression ratio leads to a higher reduction in communication time. In this paper, we have used JPEG as our compression technique to study the tradeoff between loss in quality and reduction in communication time.

4 A Framework to Support Fast Image Retrieval in a Wide Area Network

4.1 Parameters Involved in Retrieval

We have identified three different phases in image retrieval in a wide area network:

- Accessing the data object on the remote computer (time taken: t_a)
- Transmission of the data object over the network (time taken: t_t)
- Display on the local computer (time taken: t_d)

$$response\ time = t_a + t_t + t_d$$

The following subsections contain a description of the parameters which influence each phase. The first five parameters are dependent on the resources available and the user has no direct control over their values. The values for the last two parameters can be specified by the user. The parameters image quality and response time are the two parameters that can be changed so that specifications can be complied with. Image quality also can be changed automatically by the system.

4.1.1 Transmission Distance

The Internet spans the entire world. A user may want to retrieve data from halfway across the world. The time taken for a user in W. Lafayette, Indiana, to retrieve an image file from a site in California is more than the time taken for the same user to retrieve an image file from a site in Maryland. The user might be satisfied with a lower quality image to reduce the time delay when retrieving from California but while retrieving from a site close by he might prefer the perfect quality image. Network distance is one of the important factors in t_t and contributes a major portion of the response time

4.1.2 Image Attributes

Lossy compression techniques reduce the size of the file by losing some information. How much information is lost determines the final size of the file. Different compression techniques are suitable for different images depending on the kind of the information the image contains. As described in Chapter 3, JPEG works well on some material like photographs, naturalistic artwork, and but not so well on certain other material like lettering, simple cartoons, or line drawings [3]. GIF is better for such images and often compresses them more than JPEG can. Even in one compression scheme, some images compress better than others - i.e; they can lose more pixel information without losing semantic information. When subjected to the "eyeball" test (using the human eye to distinguish between two images), they look the same as the original. For example, because of the cube effect in JPEG compression, for images with predominantly a single color with different shades anything beyond 30% distorts the image. Some images do not lose any semantic information in greyscale but some look completely different. Depending on the nature of image, we can assign attributes to the image and these attributes can determine the amount of compression the image can tolerate, which format is suitable and whether greyscale is adequate.

4.1.3 Time of Day

Internet traffic varies very much with the time of day [9]. Traffic at night is less and a user retrieving images at that time, might be able to get the full size high quality image with an acceptable delay. Our experiments have shown that image transmission in the day took more time than at night because of network traffic during the day. When the remote site is halfway across the world in an entirely different time zone then part of the network will be in the light traffic time slot, and part will be in the heavy traffic slot.

4.1.4 Computation Power

If images are compressed online and then transmitted then the computation power at the sender site will influence the display time. For instance, a SPARC 10 can compress a 400K GIF file to 10% JPEG file in 2.9 secs whereas a SPARC 1 takes nearly 15 secs for the same file.

The machine at the receiver site also has an influence on the display time. Display of the same 400K file at 10% JPEG compression takes 4.8 secs on a SPARC 10 and 4.4 secs on a SPARC 1. Display of the same file in a GIF format on the SPARC 10 takes only 2.8 secs. For display, format is more important than the size. But GIF files are larger and would take more transmission time. The decision as to whether the image should be compressed or not before transmission and what format should be used should include the factors of the computation power at both the sender and receiver sites.

The other factor is that different compression schemes take different times for compression and decompression. A quick and dirty compression/decompression scheme may be useful in performing lossy compression.

4.1.5 Physical Media

High speed network technologies like ATM, FDDI, will reduce transmission time. In a wide area distributed environment like the Internet, the user has typically no control over the kind of network the image will be transmitted on but if the type of network is known (as in a LAN) then that will influence the decision of the size of image that should be transmitted.

4.1.6 Storage

To save time and reduce load on the remote machine, the alternative to online compression is to store the original image and copies at all the different quality levels. Current technology has made storage very cheap but still it is not free. It may not be viable to store all the different levels of compression of a file. If a certain quality level is used only very rarely then it should not be stored. Online compression depending on the other parameters has to be done.

4.1.7 Quality level (user control)

Different users are satisfied with different levels of quality. An earth scientist looking at a NASA image may want all the details of all the shades without any distortions whereas a high school student might be satisfied with bare outlines which give him an idea of what is contained in the image. Based on a level specified by the user, the image can be processed to lose information, thus reducing display time. A particular class of user may not be a frequent user. If that user

requests x% compression, that compressed file should not be stored but produced online since the storing an infrequently used file is an inefficient usage of storage space.

The quality level need not be specified by the user always as a percentage. For example, there can be levels the user can choose from: perfect, good, mediocre, bad and intervals of quality levels can be associated to them by the system.

4.1.8 Response Time (user control)

The user can decide the response time he can wait for. If the response time is high, he will get a better quality image. The response time might not be enough to transmit an image of the requested quality. This leads to a conflict in which case either the user has to revise his response time requirement or not retrieve the image.

4.2 Preserving Semantics with JPEG

JPEG is the compression scheme we have chosen to use for measuring access times in our experiments. The objective is to compress an image so that the semantics are not lost. We have seen that JPEG does not do a very good job when images have smaller number of colors or sharp edges. Consequently, compression ratio of a file is dependent on its content. A file with a large number of colors can have 5% compression whereas a file with a smaller number of colors can have 30% compression. All images are not treated in the same way.

Depending on the semantics of the image, a very low quality image is sufficient to pass the eyeball test. For such an image we have the added advantage that JPEG itself does a good job. So depending on the semantics of the image, the amount of compression (10%, 30% and so on) should be decided. We propose that the following attributes should be associated with the image: number of colors, number of sharp edges, and amount of space with all pixels exactly the same color. Based on these attributes, the quality level for that image can be determined. Different compression techniques will be influenced by different attributes. Depending on the compression technique used, they should be varied.

Let us consider the nineteen files we have used for our experiments. The images are included in the appendix. We found that when compressed to 10% JPEG level, the images could be grouped into four classes:

- Perfect: The compressed image is indistinguishable by the human eye from the original. Example: ast_spray.
- Partial Loss of Information: Some minute details like small dots, small, thin lines are lost in the image. Example: earth1
- Complete Loss of Regions: Some regions in the compressed image are missing. Also, some colors are extra bright. Example: hubble.costar
- New Regions are Introduced: New regions, which were non-existent in the original image appear in the compressed image. Example: mars

The above observations indicate that all images cannot be compressed to the same level and uncompressed so that they look the same to the human eye. Some images can tolerate more

compression than others. If the system provides four levels for the user to choose from - perfect, good, mediocre, and bad, then all these images at these levels will not correspond to the same four levels of quality percentage. An image may be perfect at 10% level and another may be bad at that level. Thus "how much can an image be compressed" is an important attribute of the image. If an image cannot tolerate much compression, then for many users response time for that image cannot be significantly reduced.

4.3 Experiments to Measure Response Time

We had observed during our use of Mosaic, Netscape, and other browsers that the response time varies with network distance, with the time of usage, and with the workstation we were working from. Some remote sites were faster than others. Some workstations were faster than others. Some times of day were better than others to browse the Internet. This gave us the motivation to conduct experiments to get a precise estimate of the time involved in the different phases of fetching an image from a remote site. We wanted to use this data to study how, if any, phases of the the retrieval process could be speeded up by trading off some parameters for others. With this objective in mind, we conducted the following experiments:

1. Measure the time taken to transmit an image file over an network (t_t). This gives us:
   - Relationship between network distance and t_t
   - Relationship between time of day and t_t
   - Relationship between size of file and t_t
2. Measure the time taken to convert GIF files to different quality JPEG files and from one quality JPEG file to another (a part of t_a). This gives us:
   - Relationship between sender computation power and t_a
3. Measure the time taken to display the different format (GIF and JPEG), different quality (10%, 30% JPEG etc.) images on different machines (SPARC 1, SPARC 5, SPARC 10) (t_d). This gives us:
   - Relationship between receiver computation power and t_d

4.3.1 Experimental Setup

We conducted the experiments in the Raid Laboratory at Purdue University. The machines in the laboratory are connected through a 10Mbps Ethernet. We used three SPARCstations – raid8, a SPARC 1; raid4 a SPARC 5; and pirx, a SPARC 10. We used a millisecond resolution clock to measure the time in all our experiments.

4.3.2 Input Parameters

Nineteen files with sizes ranging from 7K to 400K were chosen for the experiments. The criteria of selection was representation of a particular image type like shades, lines etc. They were chosen so that their sizes spanned the interval of 7K to 400K. A description of the files is given below:

- earth-round.gif: Sharp contours, green on blue globe. Res: 187x158, Size: 6988 bytes
- earth1.gif: Very sharp contours, green on blue globe. Res: 160x160, Size: 7708 bytes
- gal_line.gif: Red on black, a whole line of only dots. Res: 450x450, Size: 170 27 bytes
- gal_green.gif: Green on green, lots of dots, striations of colors. Res: 384x330, Size: 29668 bytes
- comet.gif: White eye, blue tail, tail fades into background. Res: 512x480, Siz e: 35543
- mars.gif: Huge circle of light brown shades. Res: 340x340, Size: 60379 bytes
- surface.gif: Sloping surface of white and blue, some sharp contours, some shades. Res: 550x450, Size: 74058 bytes
- jupiter.gif: Huge circle of red and yellow shades, yellow text on black. Res: 710x765, Size: 80385 bytes
- gal_blue.gif: Blue on blue, some dots. Res: 607x373, Size: 97835 bytes
- hubble.costar.gif: Shades of concentric red, orange, yellow colors, some shading, some text. Res: 566x384, Size: 104365 bytes
- earth_detail.gif: Blurred contours, some text, pink color, black background, w hole globe. Res: 1152x864, Size: 114323 bytes
- eclipse2.gif: A huge number of red shades. Res: 784x630, Size: 135701 bytes
- 4gal_red.gif: Bright colors (red, orange), white spots, black and white dots, some shading. Res: 441x400, Size: 153634 bytes
- sf.gif: Sharp boundary contours, blue, white and red colors. Res: 500x500, Size: 175405 bytes
- ast_spray.gif: Black background, lots of small particles. Res: 701x659, Size: 205747 bytes
- mitwave1.gif: Orange and white shades, delicate, multicolored ridges. Res: 1024x1024, Size: 236199 bytes
- earth_highres.gif: Blurred contours, some text, blue color, black background, a part of the globe. Res: 1152x864, Size: 279786 bytes
- text+image.gif: Text, many dots, subtle shading. Res: 936x867, Size: 406851 byt es
- eclipse1.gif: A huge number of orange and yellow shades. Res: 1280x1024, Size: 486430 bytes

4.4 Experiment 1: Transmission over the Network

4.4.1 Statement of the Problem

The purpose of this experiment was to measure the time taken to transmit image files of different sizes from Purdue to five chosen remote sites across the Internet. This would give us a measure of the amount of overhead due to communication in a wide area network. Intuitively, one can observe that the transmission time increases as the number of hops the image files travels increases. We wanted to measure precisely the time taken since that would give us a measure of how much time we are saving when a lower quality image file (and hence smaller size) is used.

4.4.2 Procedure

It is very difficult to have computer accounts in more than one administrative domain [9]. Computer accounts are mostly restricted to ones organization and that will be in one geographical location. We needed to conduct experiments across the Internet. Since we do not have an account on the remote computer, we cannot receive data objects and record timestamp information. To measure the transmission times from our computer X to a remote computer Y, we used a TCPecho program to send the image file from X to Y, echo off Y and return to X. This gave us a measure of the round trip time without using an account on Y. Our TCPecho program used a TCP client to echo a file using port 7 on the specified host.

Each trial consisted of 100 repetitions and we had three trials to get a total of 300 repetitions.

Raid8, a SPARC 1 machine and a part of the Computer Science department subnet was the machine which we primarily used for experiments. We chose the following five remotes sites. The number in braces gives the number of hops on the days of the experiments.

1. Pirx.cs.purdue.edu: A machine on the same LAN as Raid8. (Number of hops = 1)
2. Atom.ecn.purdue.edu: A machine on the engineering network of Purdue. This machine is on the same MAN as Raid8. (Number of hops = 4)
3. Merope.cs.buffalo.edu: This machine is in New York. (Number of hops = 19)
4. Ironweed.cs.uiuc.edu: This machine is in Illinois (Number of hops = 19)
5. Lanai.cs.ucla.edu: This machine is in California. (Number of hops = 22)
6. Bovina.cs.utexas.edu: This machine is in Texas. (Number of hops = 23)
7. Retriever.cs.umbc.edu: This machine is in Maryland. (Number of hops = 25)

The sites were initially chosen to represent an increasing order of network distances from Raid8 at Purdue. The experiments were conducted at various times of day: 9:00 am, 12:00 noon, 5:30 pm and 2:00 am.

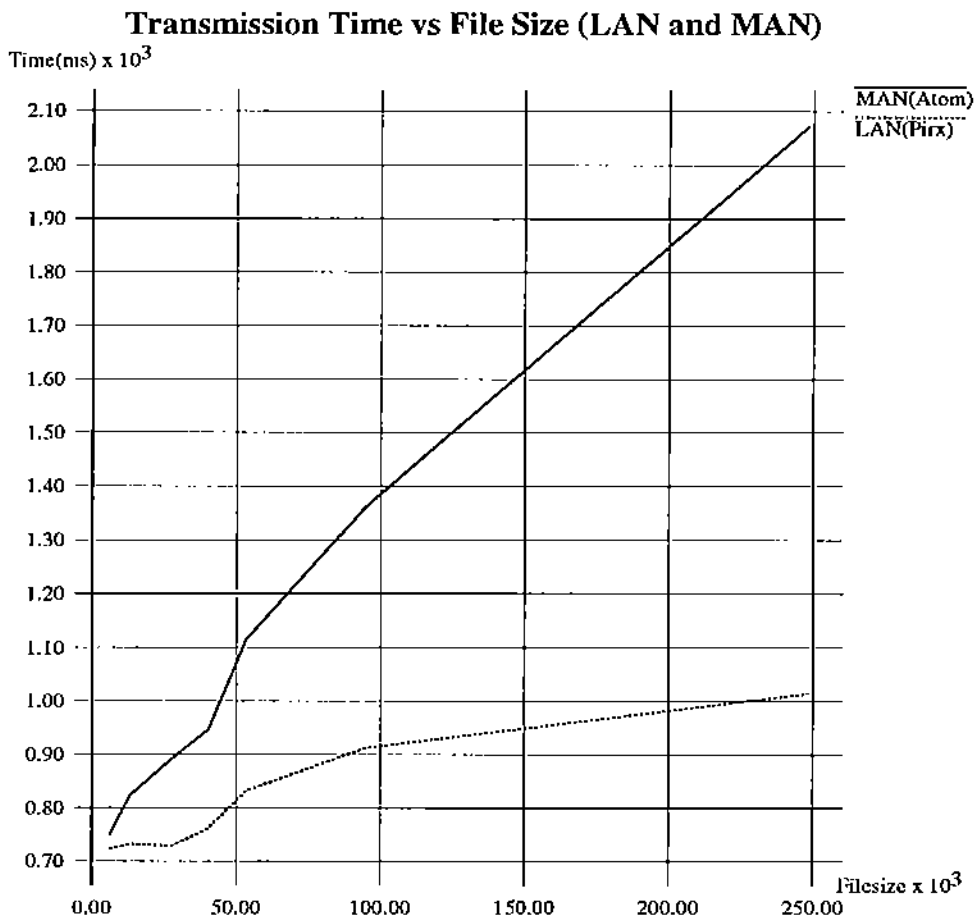

Figure 1: Variation of Transmission Time with File Size in a LAN and MAN

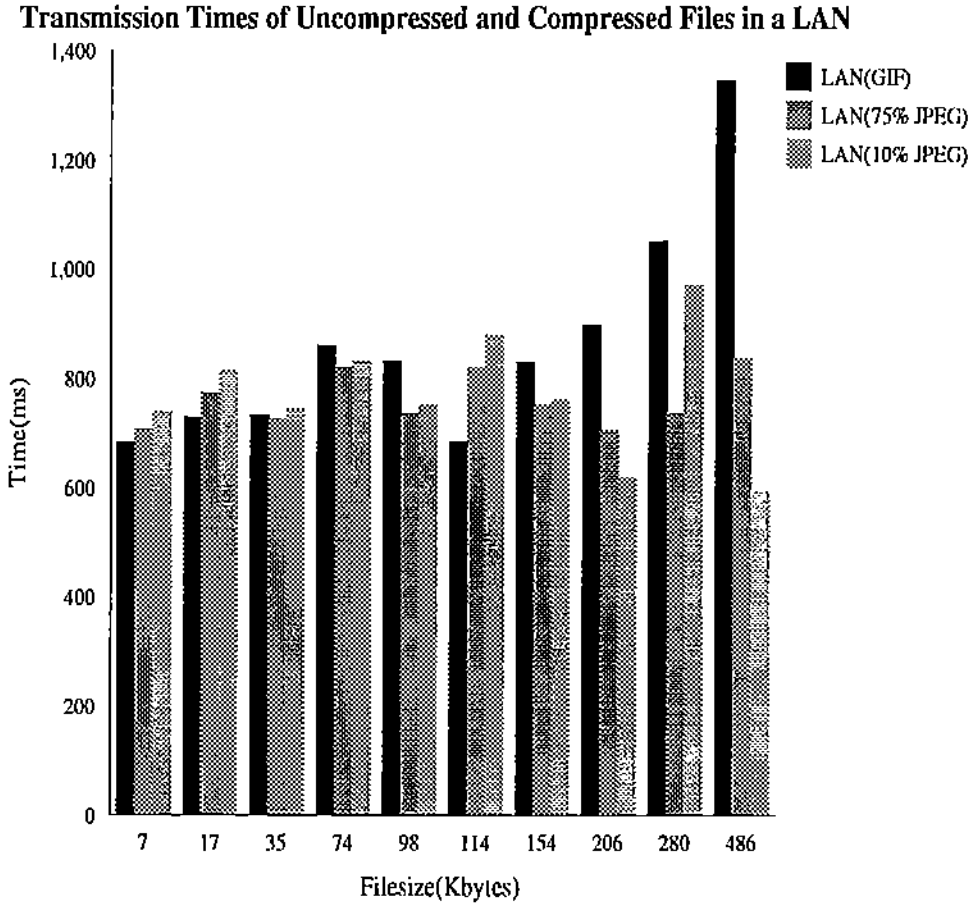

Figure 2: Variation of Transmission Time with File Quality in a LAN

4.4.3 Experimental Results

Figure 1 shows the round trip transmission time vs. filesize for the sites in the LAN and MAN. From the human perspective, in a LAN, the difference in round trip time between the smallest and largest file is not significant. Figure 3 shows a graph of round trip time vs. filesize for the five different remote sites. This shows how much network distance influences transmission time (t_t). The curves rise more sharply than in the case of a LAN. Figure 4 shows the variation of round trip time with time of day. The experiments were conducted at 12:00 noon and 2.00 am using Bovina (in Texas) as a remote site. Figure 2 is a bar graph comparing the transmission times for uncompressed (GIF) and compressed (75% JPEG and 10% JPEG) files in a local area network. Except for the larger files, there is not much difference between the three files. The X axis gives the file number assigned to the nineteen files. This was because the file sizes did not have the same order when they were sorted for the three different quality levels. For example, the file 4gal_red is smaller than the file surface at the 75% quality level, but greater than surface at the 10% quality level. Figure 5 shows the variation of round trip time with respect to distance for two different quality files (10% and 75%) in a wide area network (the remote site used for collecting this data is in California).

4.4.4 Discussion

Network Distance and Response Time: As the network distance between the user and the remote site where the image is being retrieved from increases, the response time increases (see Figure 3). The figure also shows that as the file size increases, the response time increases. The curve rises more sharply when the receiver is a remote site in a wide area network. When images are retrieved from a site in a LAN or MAN the difference between the largest file and the smallest file in our experiments is only one second. From a human perspective, this is not a significant difference. Hence only beyond a certain network distance does compressing a file achieves a significant saving in response time.

We can see that for small files with size less than or equal to 10K, the difference in transmission times between a site in a LAN and a site in a WAN is only a second (Figure 3). The difference in transmission times between two such sites increases as the size of the file increases. This is clearly illustrated in Figure 6 which shows the transmission times for the LAN site, MAN site and two WAN sites. This indicates if a file size is less than a certain threshold, the file can be retrieved as it is whatever the network distance. Large files have to be reduced in size by trading off quality. The reduction in response time increases as the network distance increases making the tradeoff worthwhile.

Time of Day and Response Time: As can be seen in Figure 4, The response time is lower when images are retrieved at certain times like 9.00 am and 12.00 midnight. 2.30 pm and 5.30 pm seem to be heavy traffic times for the Internet, as evidenced by the higher response times. If the image is being retrieved at a time when the Internet traffic is heavy, tradeoff in quality may not be required. If traffic is heavy, as in the daytime, a tradeoff in quality is worthwhile to achieve lower response time. At a low traffic time, the threshold chosen for network distance and file size will be higher.

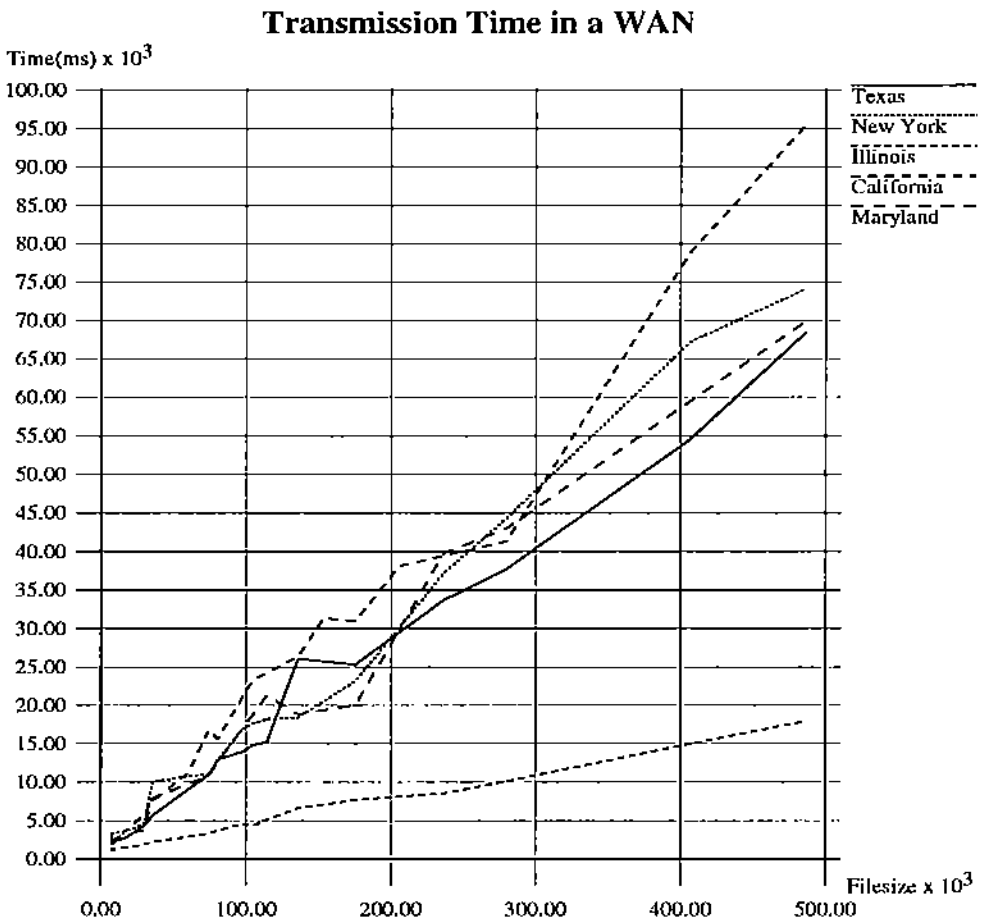

Figure 3: Variation of t_t with File Size

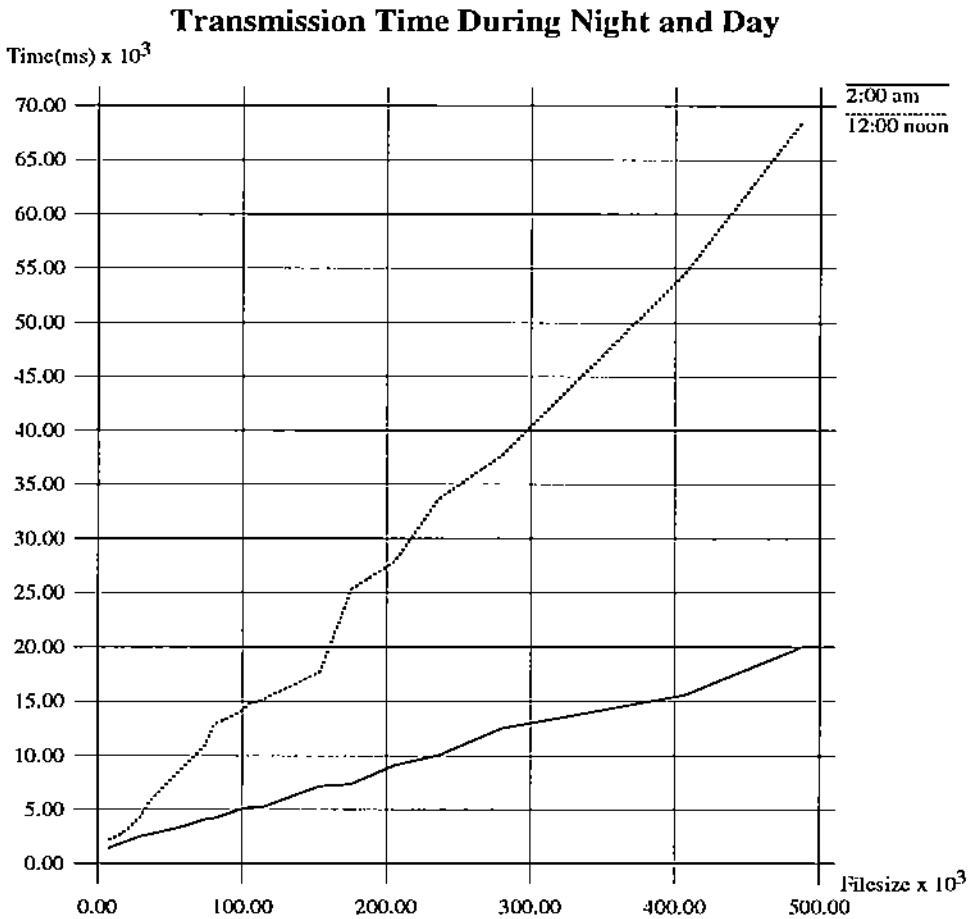

Figure 4: Variation of t_t with Time of Day

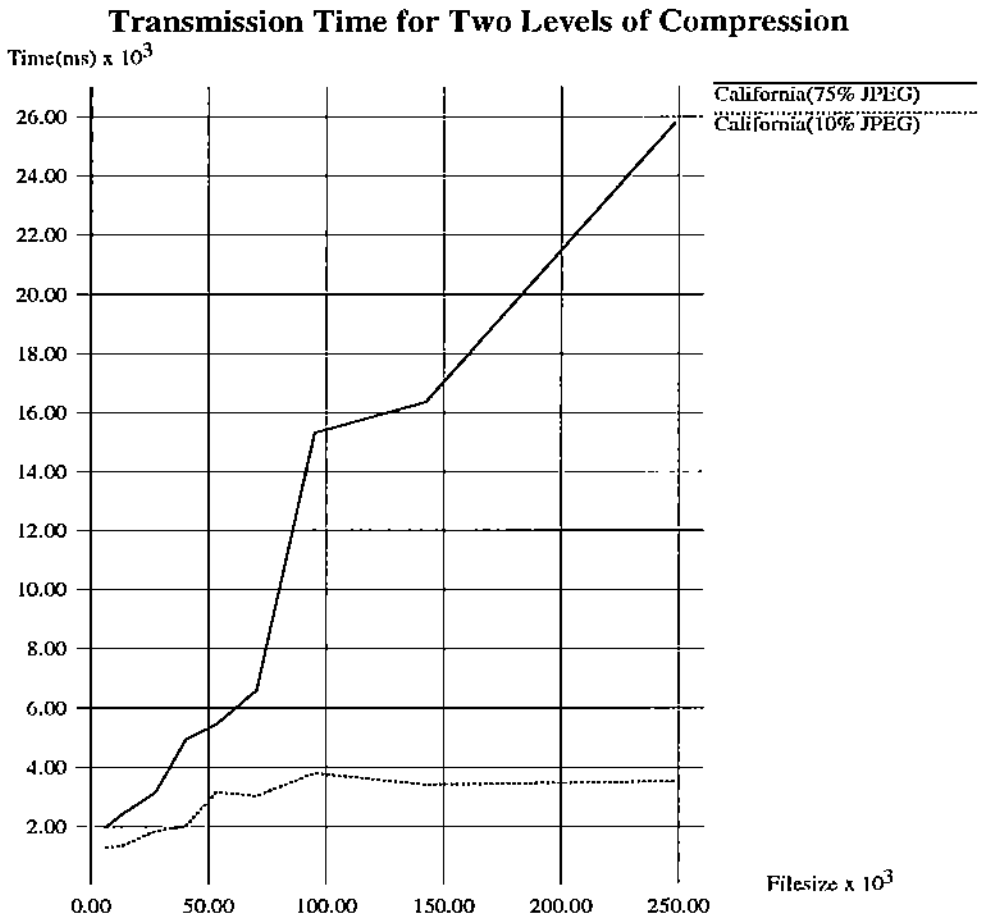

Figure 5: Variation of t_t for Different Quality Files

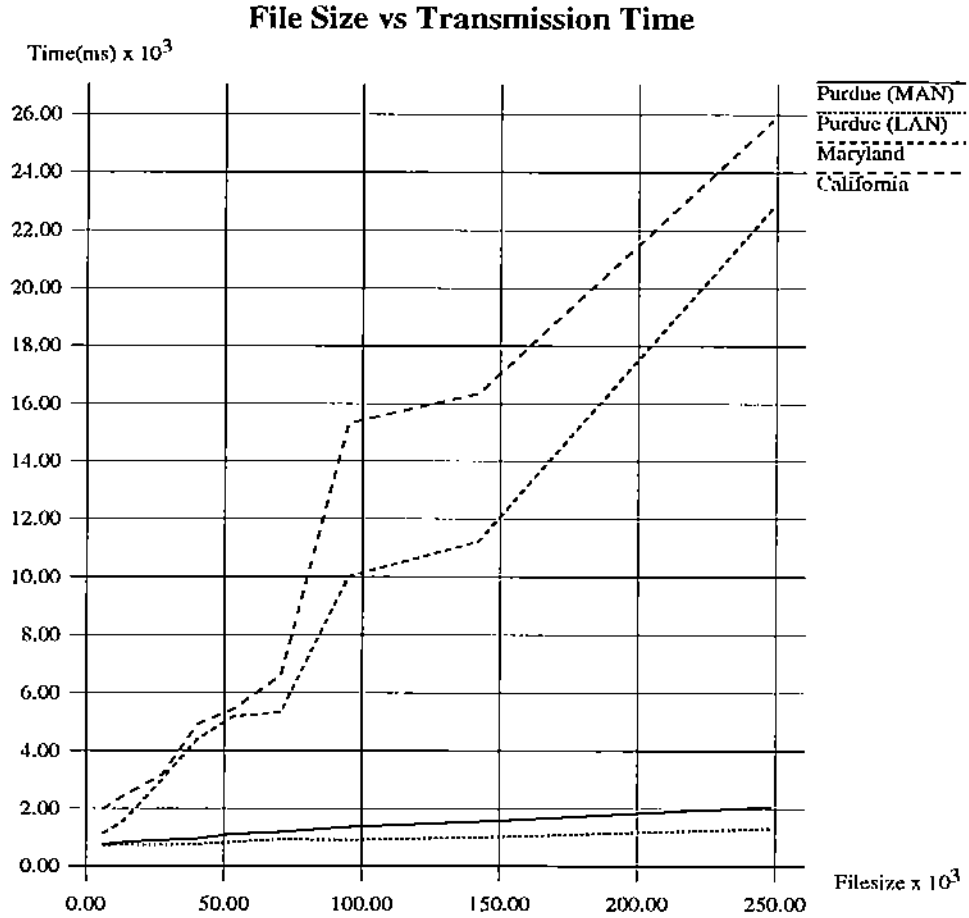

Figure 6: Variation of t_t in LAN and WAN Environment

4.5 Experiment 2: Conversion from GIF to JPEG Format

4.5.1 Statement of the Problem

The purpose of the experiment was to measure the time to perform lossy compression to a certain quality level for different types of image files. The compression method we chose for our experiments was the JPEG compression scheme. GIF was the format we chose to represent the uncompressed form. This was because JPEG is a standard and GIF and JPEG are two of the most popular formats for images in use today. We used three machines for conducting the experiments – raid8 (SPARC 1, black & white), raid4 (SPARC 5, color), and pirx (SPARC 10, color).

4.5.2 Procedure

We used the free JPEG software developed and distributed by Independent JPEG Group (IJG). They conform to the ISO JPEG standard. The software, written in C, implements the JPEG baseline and extended-sequential compression processes. It includes a set of library routines to write and read JPEG image files, and provides "cjpeg", an application to use the routines to perform conversion from other formats to JPEG. We used a shell script to perform this conversion 100 times for each of the nineteen files. This shell script was executed three times bringing the total number of repetitions to 300 for each file. The experiment was performed at night to ensure low load on the system and the system was periodically checked to see that no other major process was running.

4.5.3 Results

Table 1 gives a table of the time taken to convert a GIF file to 10% JPEG file on three different machines for four image files (sizes 400K, 200K, 100K, 7K). Figure 7 shows a graph of the same data for nineteen different files on the three machines.
The time taken for this conversion is not directly proportional to file size. We see that a 153 K GIF file takes less time to convert than a 135 K GIF file.

Filesize	400K	200K	100K	7 K
Sparc 1	2892.910ms	1870.690ms	999.820ms	358.250ms
Sparc 5	4138.700ms	2511.190ms	1514.350ms	429.150ms
Sparc 10	14547.990ms	9055.750ms	4213.340ms	838.000ms

Table 1: Time Taken to Convert a GIF File into 10% JPEG File

4.5.4 Discussion

By looking at Figure 7 one can observe that different images take different time to compress to a certain quality level. There does not seem to be any clear correlation between filesize and the time taken to compress the file using a JPEG compression algorithm. The time taken

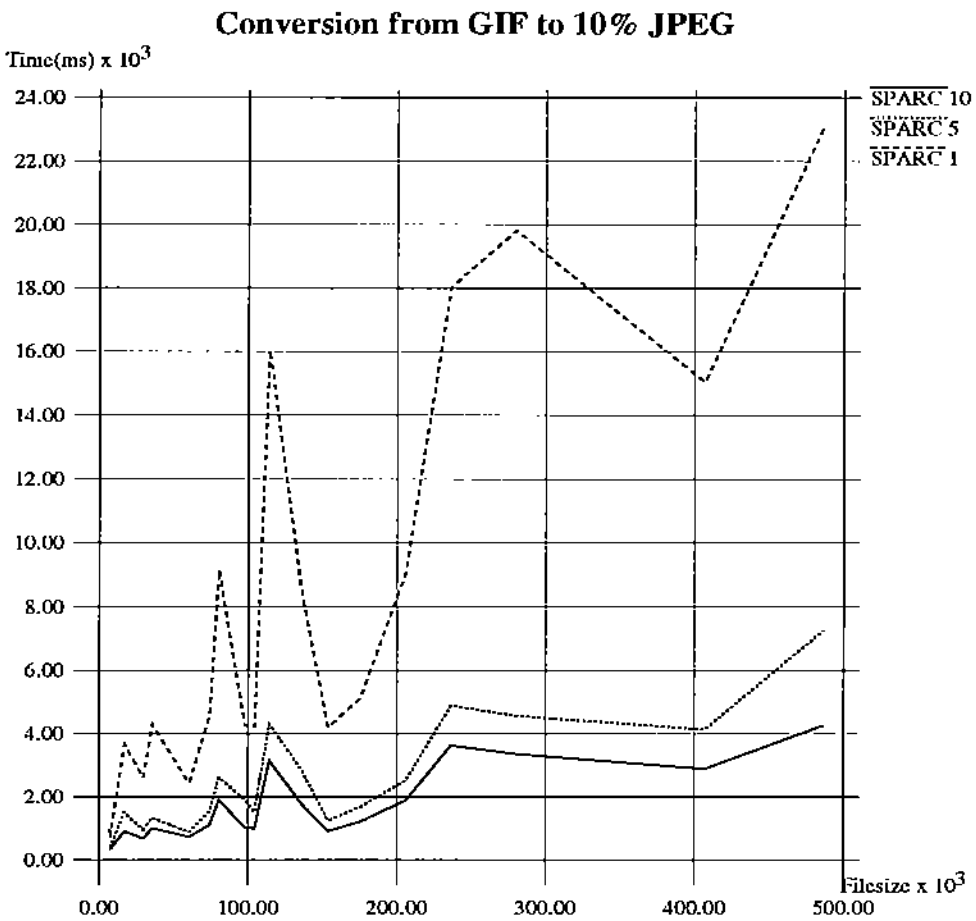

Figure 7: Conversion Time for GIF to 10% JPEG

for compression depends on the semantics of the image like the number of colors, number of shades, amount of detail and so on. All files do not reduce their size by the same percentage after compression. That depends on the semantics of the image file too. Table 2 shows the nineteen file sizes before and after compression and the percentage of the compressed file when compared to the original file. If the compressed file is a large percentage of the original file then the transmission time will not be significantly reduced. In such a case transmitting the uncompressed file may be the better option.

Filename	GIF Filesize	10% JPEG Filesize	% of GIF File
earth-round	6988	2297	32.87%
earth1	7708	1968	25.53%
gal_line	17027	5254	30.86%
gal_green	29668	7757	26.15%
comet	35543	7572	21.3%
mars	60379	3764	6.23%
surface	74058	7857	10.61%
jupiter	80385	16298	20.27%
gal_blue	97835	5456	5.58%
hubble.costar	104365	10374	9.94%
earth_detail	114323	25646	22.43%
eclipse2	135701	10776	7.9%
4gal_red	153634	9822	6.3%
sf	175405	11977	6.83%
ast_spray	205747	20348	9.9%
mitwave1	236199	22426	9.49%
earth_highres	279786	30520	10.91%
text+image	406851	34863	8.57%
eclipse1	486430	22493	4.62%

Table 2: Comparison of Original (GIF) and Compressed (10% JPEG) File Sizes

4.6 Experiment 3: Display of GIF and JPEG Files

4.6.1 Statement of the Problem

The purpose of this experiment is to measure the time taken to decompress and display different image files. Image formats have varying degrees of complexity and the time taken to decompress and display, which we call display time, varies accordingly. Display time depends not only on the image format used but on the semantics of the image as well.

4.6.2 Procedure

We used the XV shareware software (version 3.01) developed by John Bradley to display several format images in a X window environment. XV code is freely available. We modified the code

to note the time before beginning processing of an image to be displayed, displayed the image, killed the image and noted the time again. The difference in the time stamps gave us the time taken to display the image. This time did not include the time taken to load the XV software. Each trial consisted of displaying the image 100 times. We conducted three trials resulting in 300 repetitions. Again, the experiment was performed at night to ensure low load on the system and the system was periodically checked to see that no other major process was running. We measured the display times on three machines as in experiment 2 – raid8 (SPARC 1, black & white), raid4 (SPARC 5, color), and pirx (SPARC 10, color). The formats displayed were GIF, JPEG and 10% JPEG.

4.6.3 Results

As with measuring conversion times, the display time was measured on the three machines SPARC 1, SPARC 5, and SPARC 10 for the nineteen files. The files displayed were GIF, JPEG and 10% JPEG. Table 3 gives the display times for JPEG files. Table 4 gives the time taken to display GIF files. Figure 8 shows how display times vary for different file sizes and different formats. The X axis gives the file number assigned to the nineteen files. We can see from the figure that though GIF files in general take lesser time to display than JPEG files, for two files (eclipse1 and mitwave1), JPEG is faster than GIF.

Filesize	400K	200K	100K	7 K
Sparc 1	16065.67ms	11736.33ms	5292.78ms	1946.11ms
Sparc 5	7009.44ms	4901.33ms	3124.78ms	3157.78ms
Sparc 10	4837.0ms	3592.0ms	2183.67ms	1429.56ms

Table 3: Time Taken to Display JPEG Files

Filesize	400K	200K	100K	7 K
Sparc 1	12962.56ms	8193.11ms	4410.11ms	1595.33ms
Sparc 5	4195.44ms	2880.89ms	2409.33ms	1583.44ms
Sparc 10	2836.56ms	2025.56ms	1737.11ms	1170.22ms

Table 4: Time Taken to Display GIF Files

4.6.4 Discussion

Image Format and Display Time: Different formats for image files take different times to display. In general, GIF files are quicker to display than image files [3] because of a simpler decoding algorithm. But GIF files are larger than JPEG files and hence will take longer time to transmit.

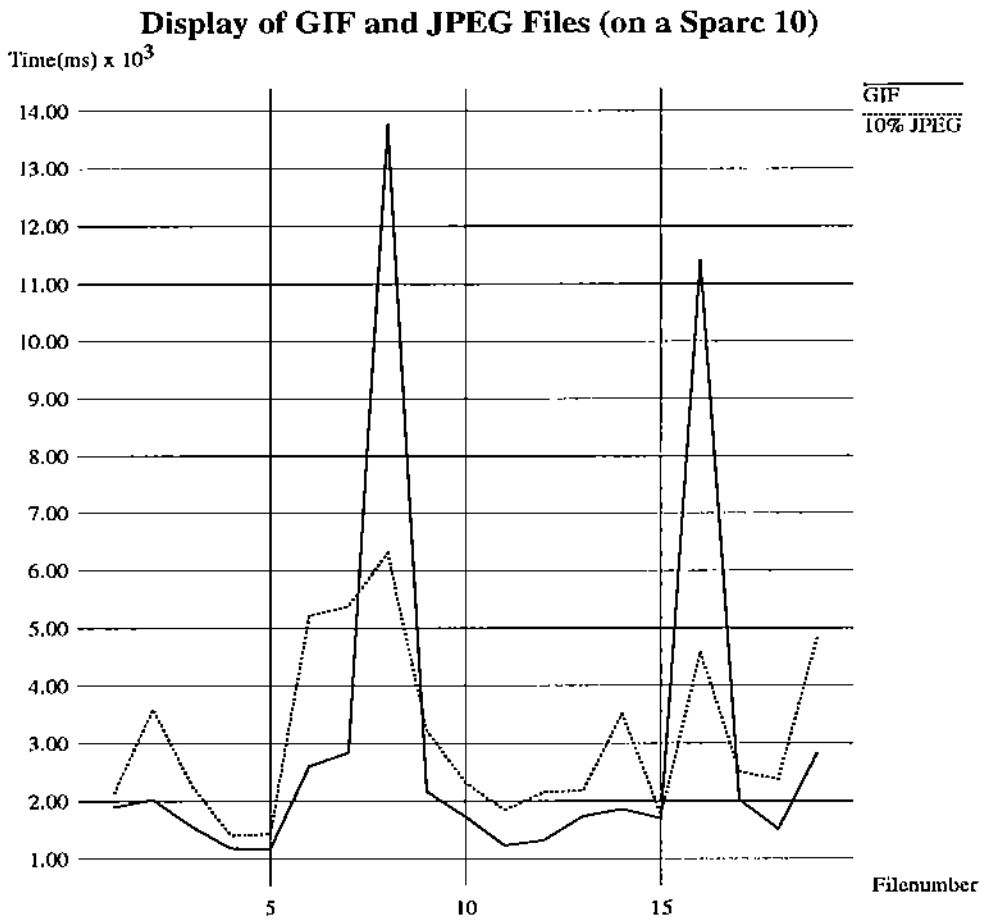

Figure 8: Display Time for GIF and JPEG Files

If the distance is small, and the GIF file takes only a fraction more than the JPEG file then GIF can be used since GIF files are usually faster to display. GIF files are not always faster to display however. Depending on the contents of the image, a GIF file can be slower to display than the JPEG format of the same file. This should be an attribute of the image. If the time taken to compress, transmit, decompress, and display is less than the time taken to transmit the uncompressed file, then the file should be compressed and transmitted. The last phase of decompression and display might outweigh the advantage of compressing and transmitting the image which reduces the response time in the first two phases.

4.7 Reducing Response Time

The data we have collected so far can be used to decrease the time taken by some of the phases which constitute the total response time. Depending on the parameters involved in the retrieval, different modes of retrieval should be adopted instead of an uniform mode for all images.

4.7.1 Dependencies between User Specified and Other Parameters

Transmission time depends on:

- Network Distance
- Time of Day
- Image Quality (file size)

Access time depends on:

- Computation power of the sender machine - where the image file is being retrieved from

Display time depends on:

- Computation power of the user machine which receives the image file

Image quality can be changed by both the system and the user. The more image quality is reduced, the lesser is the transmission time. Depending on the attributes of the image, there is a threshold beyond which the image can't be compressed without losing semantic information. Beyond this threshold image quality cannot be controlled by the user.

Time of day can be changed by the user. The user might prefer to retrieve an image at night when there is less traffic. If cost is associated with each retrieval, then retrieval at night can be charged less than retrieval during the day, in a way similar to telephone companies.

The other factors network distance, computation power of sender machine, and computation power of receiver machine are beyond the control of the user.

4.7.2 Specification by the User

The user can specify two parameters – response time and quality level. Based on these parameters the system decides the image format, and quality to be transmitted that will minimize response time. In the following subsections we suggest steps that can be carried out to arrive at the appropriate image format and quality level. First we consider the case where the user

specifies response time and image quality (quality level) is left to the system. Next we consider the case where the user specifies the quality level (what quality of image he requires) and the response time is left to the system.

Objective: To minimize $(t_a + t_t + t_d)$

t_a depends on computation power of the sender machine.
t_t depends on distance and time of day.
t_d depends on computation power of the receiver machine.

Let q represent quality level.
Let t_a^q be the time taken to perform lossy compression on the image so that $q\%$ information is retained.
Let t_t^q be the time taken to transmit the image after it is compressed to the specified quality q.
Let t_d^q be the time taken to display the compressed image.

User Specifies Response Time

The user specifies that he wants the file to be retrieved in time T. The objective is to give him the best quality image possible within the time T. Given time T, if the only quality of the image possible falls below that specified by the image attributes, then there is a conflict between the user's requirements and the system's possibilities. The user has to revise his requirements or be satisfied with an image which has completely lost its semantics.

$q = 100\%$
While $(t_a + t_t^q + t_d^q \geq T)$
    $q = q - \delta$
WhileEnd
Transmit image with this q

User Specifies Quality Level

As mentioned above, some users require a very high quality image which they will use for other purposes, some are satisfied with a general idea of the image is like. When the user specifies a quality q he is interested in, the system should find out the fastest way to get an image of that or greater quality on the user's screen. If it is cheaper to get a top quality image without any lossy compression being done on it, then that should be preferred over the scheme of compressing the image and losing some information, transmitting it, and reconstructing the image at the user's local site.

q = user specified quality level
$q100$ = 100% quality

If $(t_a^q + t_t^q + t_d^q \leq t_a + t_t^{q100} + t_d^{q100})$
then *compress and transmit*
else *transmit as is*

The amount of compression that can be done depends on the attributes of the image. Images can be categorized by whether or not they require color, whether the images have sharp detail or less than sharp detail, whether there are shade variations and blocks with the exact same color and so on.

Steps to be Followed before Transmitting an Image

1. If the size of the file is less than a certain threshold, transmit without any processing
2. If the network distance is less than a certain threshold, transmit without any processing
3. Identify from the attributes of the image how much lossy compression it can tolerate. If more data is lost, there will be loss in semantic information. Let this be Q_{image}
4. Note the time of day. Perform a table lookup to get an estimate of t_t
5. Get parameters specified by user. Calculate Q_{user}, the quality which is possible (as in the previous subsections). If Q_{user} is less than Q_{image}, then there is a conflict

Time taken	Case 1: WAN	Case 2: LAN
Round trip (75% JPEG)	16.3 s	1.2 s
Round trip (10% JPEG)	3.6 s	0.9 s
Convert (75% JPEG to 10% JPEG)	2.9 s	2.9 s

Table 5: An Example: Using 10% JPEG is Sometimes Viable

We can clearly see in Table 5, that in Case 1, it is worthwhile to compress the file to a lower quality and transmit it whereas, in the second Case, it is not worth it. Figure 10 illustrates this. The five points on the X axis represent five sites: Pirx (LAN), Atom (MAN), Retriever (Maryland), Bovina (Texas), Lanai (California). We can see that for Pirx and Atom, the response time for GIF files is lower while for the other three sites response time for 10% JPEG files is lower. This gives us an idea whether a file should be transmitted in a compressed or uncompressed form based on network distance.

Figure 11 illustrates the fact when the size of the file of is small, GIF response time is lower than 10% JPEG response time. In such cases, the GIF file should be transmitted.

5 Conclusion and Future Work

The nature of image data makes it possible to lose some data without losing the semantics of the data object. Quality can be traded for a reduction in response time. As we have observed

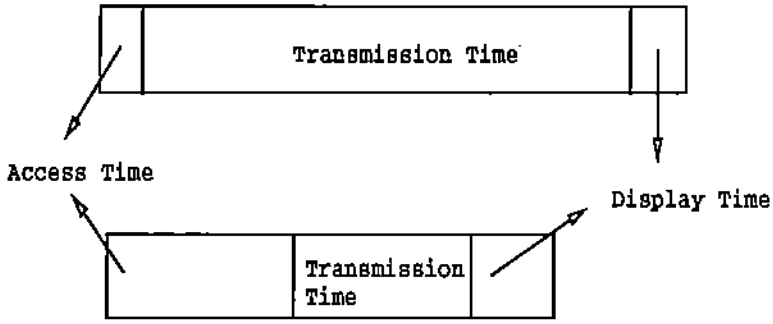

Figure 9: Increase in Access Time is Made Up by Decrease in Transmission Time

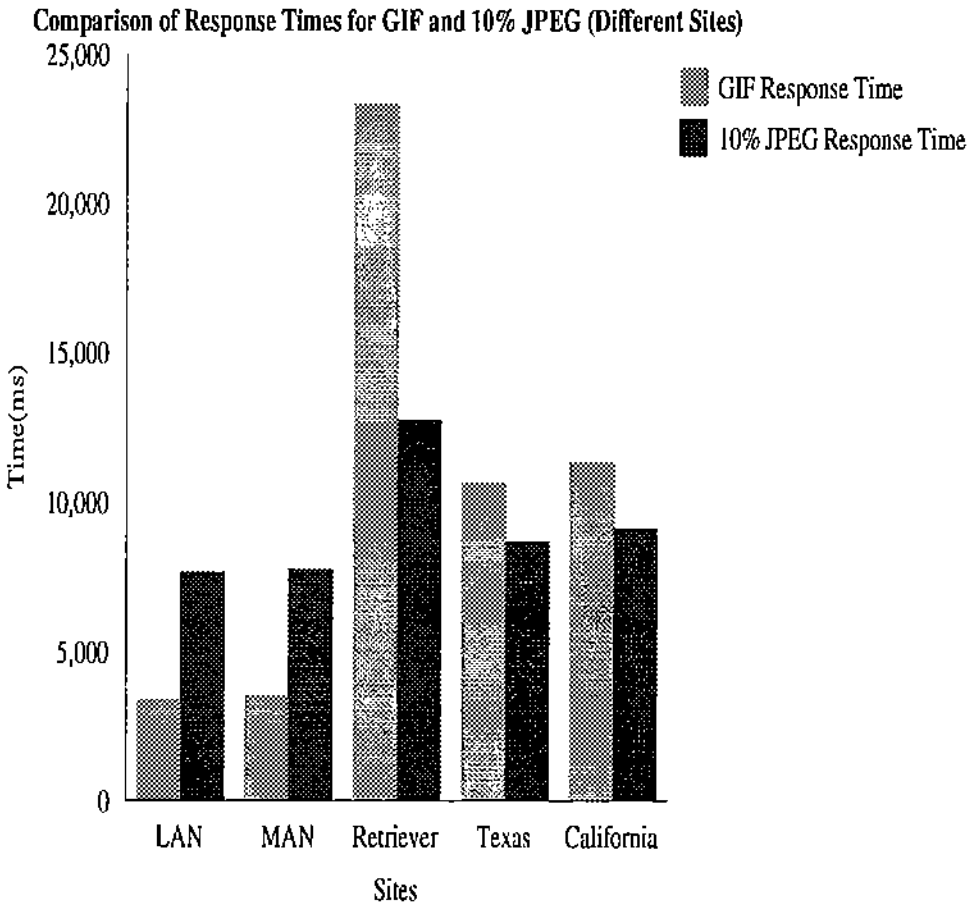

Figure 10: Transmission of 10% JPEG Files for Large Network Distance

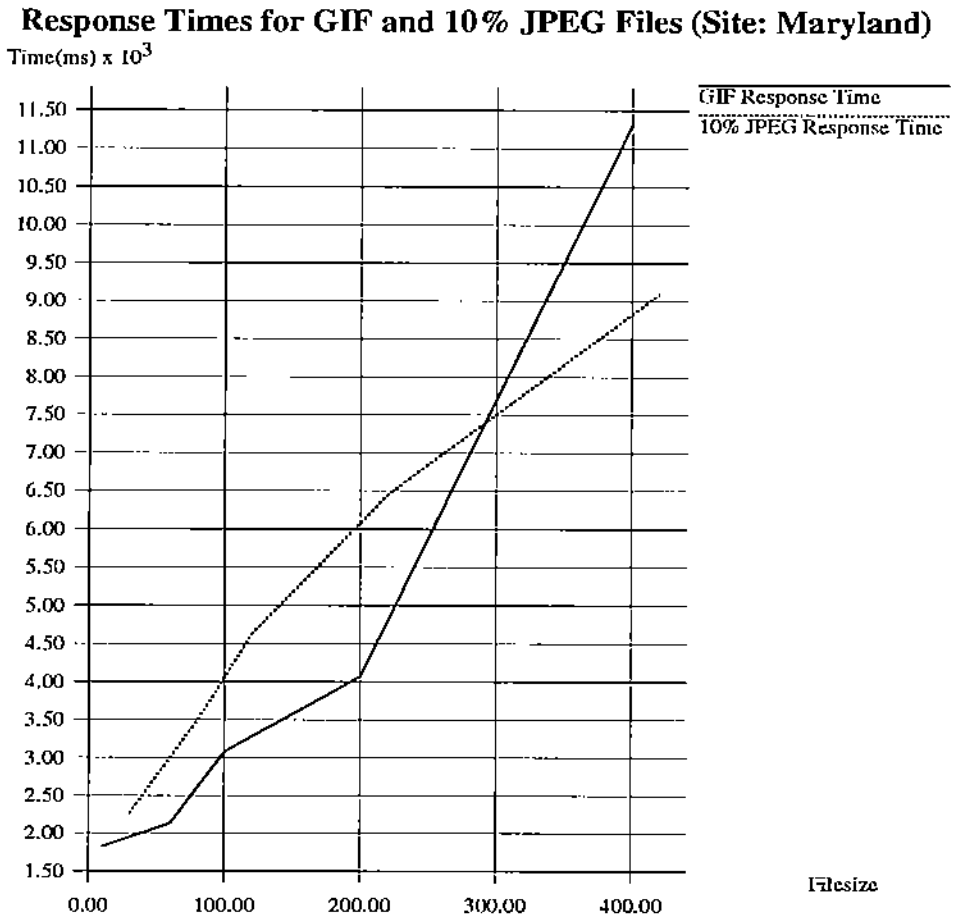

Figure 11: Transmission of 10% JPEG Files for Large File Sizes

experimentally, there are several phases in the retrieval process in which some component of response time can be reduced. Sometimes increasing the time in one phase will sufficiently reduce the time in another phase that there is a net gain. Each image and each retrieval is unique, and reduction in response time depends on several parameters. We have identified these parameters as network distance, image attributes, time of day, computation power, physical media, storage and quality level. Time of day and image quality level are two parameters that can be modified by the user to get a better response time. Computation power, network distance, physical media, and storage cannot be changed by the user and the system has to calculate the best image quality level possible under these constraints.

We plan to conduct more experiments to specify more rigorously the thresholds involved. A database of the time taken for different network distances for different file sizes can be used to calculate the best quality file to be transmitted.

We also plan to build an intelligent system that will take the task of selecting the resolution from the user. The user may not have any idea what resolution to ask for a particular image. From a study of the user's past requests, the system should be able to compute the quality level the user needs and to retrieve that image.

References

[1] L. Bergman, H. Fuchs, E. Grant, and S. Spach. Image rendering by adaptive refinement. *Computer Graphics*, 20(4):29–38, August 1986.

[2] R. M. Gray, P. C. Cosman, and E. A. Riskin. *Image and Text Compression*, chapter Image Compression and Tree-Structured Vector Quantization, pages 3–34. Kluwer Academic Publishers, 1992.

[3] JPEQ-FAQ. at rtfm.mit.edu. From news.answers archive.

[4] F. S. Hill Jr., S. Walker, and F. Gao. Interactive image query system using progressive transmission. *Computer Graphics*, 17(3):13–24, July 1983.

[5] D. C. Kay and J. R. Levine. *Graphics File Formats, second edition.* Windcrest/McGraw-Hill, 1995.

[6] Ken Knowlton. Progressive transmission of grey-scale and binary pictures by simple, efficient, and loss-less encoding schemes. *Proceedings of IEEE*, 68(7):885–896, 1980.

[7] R. L. Read, D. S. Fussell, and A. Silberschatz. A multi-resolution relational data model. In *Proceedings of 18th VLDB Conference*, pages 139–150, Vancouver, Canada, 1992.

[8] G. K. Wallace. The JPEG still compression standard. *Communications of ACM*, 34(4):31–44, April 1991.

[9] Y. Zhang and B. Bhargava. WANCE: A wide area network communication emulation system. In *Proceedings of IEEE Workshop on Advances in Parallel and Distributed Systems*, October 1993.